Blocking And Nonblocking In Verilog

Feb-9-2014

Blocking and Nonblocking Statements

Blocking Statements: A blocking statement must be executed before the execution of the statements that follow it in a sequential block. In the example below the first time statement to get executed is a = b followed by

```
1  a = b;
2  out_d = 0;
3  {carry,sum} = in + sum_in;
```

Nonblocking Statements: Nonblocking statements allow you to schedule assignments without blocking the procedural flow. You can use the nonblocking procedural statement whenever you want to make several register assignments within the same time step without regard to order or dependence upon each other. It means that nonblocking statements resemble actual hardware more than blocking assignments.

```
1  module block_nonblock();
2  reg a, b, c, d , e, f ;
3
4  // Blocking assignments
5  initial  begin
6    a = #10 1'b1;// The simulator assigns 1 to a at time 10
7    b = #20 1'b0;// The simulator assigns 0 to b at time 30
8    c = #40 1'b1;// The simulator assigns 1 to c at time 70
9  end
10
11 // Nonblocking assignments
12 initial  begin
13   d <= #10  1'b1;// The simulator assigns 1 to d at time 10
14   e <= #20  1'b0;// The simulator assigns 0 to e at time 20
15   f  <= #40  1'b1;// The simulator assigns 1 to f at time 40
16 end
17
18 endmodule
```

You could download file block_nonblock.v here

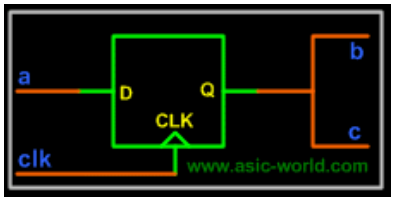
Example - Blocking

```
1  module blocking (clk,a,c);
2  input clk;
3  input a;
4  output c;
5
```

```
 6 wire clk;
 7 wire a;
 8 reg c;
 9 reg b;
10
11 always @ (posedge clk )
12 begin
13  b = a;
14  c = b;
15 end
16
17 endmodule
```

You could download file blocking.v here

Synthesis Output

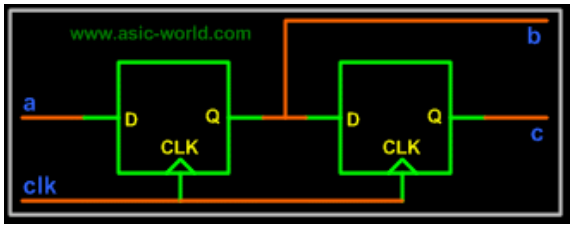


Example - Nonblocking

```
 1 module nonblocking (clk,a,c);
 2 input clk;
 3 input a;
 4 output c;
 5
 6 wire clk;
 7 wire a;
 8 reg c;
 9 reg b;
10
11 always @ (posedge clk )
12 begin
13   b <= a;
14   c <= b;
15 end
16
17 endmodule
```

You could download file nonblocking.v here

Synthesis Output



BACK HOME NEXT